**《软件形式化方法》课程教学大纲**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 英文课程名 | Software Formal Methods | | 总 学 时 | | 32 | 学 分 | 2 |
| 课程编码 | G126075 | | 理论教学学时 | | 32 | 适用专业 | 软件工程 |
| 课程类别（请在课程所属类别栏注明选修或必修） | 通识课程 |  | 实践  教学  学时 | 实验学时 | 0 | 先修课程 | 软件工程、数据结构、程序设计 |
| 大类基础课程 |  | 上机学时 | 0 | 开课学院（部） | 计算机科学与技术学院 |
| 专业课程 | 专业选修课 | 其它 | 0 | 基层教学组织 | 软件工程教学团队 |

**一、课程简介**

该课程是一门针对计算机科学与技术和软件工程等专业的本科生专业选修课。形式化方法是软件规格和验证的方法，该课程是一门专业性要求较高、理论性较强的课程，同时将理论与实践相结合，不仅需要学习有限自动机、集合论等数学知识，以及相关形式化建模语言和验证方法，同时还需要学习使用建模和验证工具PAT。从课程地位上来说，该课程是对先修课程的深化，是一门可以将学生的前期知识转换为具体分析问题和解决问题能力的课程。

**二、教学目标**

**2.1 课程教学目标**

该课程的教学目标如下：

1. 理解和掌握有限自动机、Z语言、进程代数CSP、一阶逻辑、时态逻辑、时间自动机和MDP等常用的形式化建模和验证方法的基本知识和工作原理。

2. 能够对复杂软件系统进行建模、描述、分析和验证，并且熟悉形式化建模和验证工具PAT的使用，将理论与实际相结合，从而找出软件系统的缺陷，在实际工程问题中起到辅助作用。

**2.2 课程目标与毕业要求（指标点）对应关系**

该课程支撑以下毕业要求和具体细分指标点：

【毕业要求2】 **问题分析：**能够应用数学、自然科学和工程科学的基本原理，识别、表达、并通过文献研究分析复杂工程问题，以获得有效结论。

支撑指标点2.2：能从数学和自然科学的角度对复杂工程问题进行分析建模。

【毕业要求5】 **使用现代工具：**能够针对复杂工程问题，开发、选择与使用恰当的技术、资源、现代工程工具和信息技术工具，包括对复杂工程问题的预测与模拟，并能够理解其局限性。

支撑指标点5.4：能够掌握理论分析工具，从原理角度实现对工程问题的分析和预测，并在此过程中理解解决工程问题中的各种局限性。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 课程目标 | | 毕业要求指标点 | 教学环节 | | | | |
| 课堂授课 | 实验 | 作业 | 课堂讨论 |
| **目标1：**理解和掌握有限自动机、Z语言、进程代数CSP、一阶逻辑、时态逻辑、时间自动机和MDP等常用的形式化建模和验证方法的基本知识和工作原理。 | **指标点2.2：**能从数学和自然科学的角度对复杂工程问题进行分析建模 | | √ |  | √ | √ |
| **目标2：**能够对复杂软件系统进行建模、描述、分析和验证，并且熟悉形式化建模和验证工具PAT的使用，将理论与实际相结合，从而找出软件系统的缺陷，在实际工程问题中起到辅助作用。 | **指标点5.4：**能够掌握理论分析工具，从原理角度实现对工程问题的分析和预测，并在此过程中理解解决工程问题中的各种局限性 | | √ |  | √ | √ |

**三、课程教学内容及学时分配**

**1．理论教学安排**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 章节或知识点(模块) | 教学内容 | 学时分配 | 教学要求  (应明确教学重点、难点和教学方法) | 学生任务 | |
| 作业要求 | 其他要求(自学/讨论） |
| 1 | 第一章  软件形式化方法概述 | 1. 软件开发历史、危机和软件工程 2. 形式化方法分类和应用 3. 软件形式化内容和发展 | 2 | 重点：要求学生了解软件危机和形式化方法在软件工程中的作用，初步理解常见的形式化方法及其成功案例，并了解软件系统形式化的内容和发展。  难点：形式化方法入门。  教学方法：授课+案例演示 | 查阅形式化方法相关资料和文献，对其有更深入的认识 | 查阅形式化方法在现实生活中的成功案例 |
| 2 | 第二章  有限状态机及其扩展 | 1. 有限状态机基本概念及其复合方法 2. 有限状态机生产者-消费者系统 3. Statecharts及其状态和迁移 4. 案例分析 | 2 | 重点：要求学生了解有限状态机的概念及如何对其进行复合，熟悉有限状态机经典的生产者-消费者系统，理解扩展后的Statecharts的基本原理，并充分掌握有关案例。  难点：有限状态机的复合。  教学方法：授课+案例演示 | 课后习题2.1, 2.2, 2.6；教师补充附加习题。 | 尝试对有限状态机和Statecharts进行比较。 |
| 3 | 第三章  Z语言 | 1. Z语言概述 2. Z语言表示抽象：集合、关系及函数、序列和包、自由类型、模式 3. Z语言操作抽象：模式运算及合成、通用式函数 4. 面向对象的Z语言：类的构建与封装，类的继承、初始化及通信，类的聚合与识别 5. 案例分析 | 6 | 重点：要求学生掌握Z语言的语法表示和工作原理，能够运用Z语言对问题进行建模，并了解面向对象的Z语言，熟悉其建模方法，同时充分掌握有关复杂案例。  难点：Z语言的实际问题建模。  教学方法：授课+案例演示 | 教师补充习题。 | 自学Z语言对一阶谓词逻辑的支持和验证。 |
| 4 | 第四章  面向并发系统的进程代数 | 1. 进程代数CSP：进程表示、事件及其操作、进程复合、进程通信 2. 进程代数操作语义 3. 死锁、保活性liveness与可靠性safety验证 4. 形式化建模和验证工具PAT 5. 案例解析 | 10 | 重点：要求学生了解进程代数CSP的语法和建模方法，以及语言利用操作语义如何转化为有限状态机，初步理解各类属性的验证的含义，掌握PAT工具的使用方法并能够对其中的案例进行分析。  难点：CSP语言的综合应用。  教学方法：授课+案例演示 | 教师补充案例分析。 | 自学基于CSP进程的拒绝子集、发散以及精化关系的含义。 |
| 5 | 第五章  一阶逻辑和时态逻辑 | 1. 一阶逻辑：命题逻辑、谓词逻辑 2. 时态逻辑 3. 模态逻辑 4. 线性时态逻辑 5. 计算树逻辑 6. 模型检测：利用不同算法检测系统模型是否满足由时态逻辑等描述的性质 7. 结合建模和验证工具PAT进行案例分析 | 8 | 重点：要求学生熟悉一阶逻辑和时态逻辑的概念与术语，掌握命题逻辑、谓词逻辑、模态逻辑、线性时态逻辑和计算树逻辑的语法和语义，熟悉在此基础上的模型检测技术，结合模型检测工具PAT，会根据具体待验证的性质进行建模，并进行相应的自动化验证。  难点：如何用时序逻辑描述系统属性。  教学方法：授课+案例演示 | 教师补充案例分析。 | 自学基于不同时态逻辑的各种模型检测算法具体实现。 |
| 6 | 第六章  并发系统验证扩展：时间系统和概率系统 | 1. 时间自动机基本原理 2. 基于时间自动机的形式化验证 3. 马尔可夫链和MDP模型 4. 基于MDP模型的概率系统验证和计算 5. 结合建模和验证工具PAT进行案例分析 | 4 | 重点：要求学生了解基于时间自动机的时间系统验证方法，以及基于MDP模型的概率系统验证方法，能够基于验证工具进行相关验证，熟悉相关具体案例分析。  难点：时间和概率模型应用。  教学方法：授课+案例演示 | 教师补充案例分析。 | 自学扩展了时间和概率属性的进程代数CSP。 |

**四、考核方式及成绩评定方式**

该课程总成绩分为平时成绩和期末成绩两部分，分别占50%。平时成绩主要考核学生书面作业。任课教师每个知识模块都须布置具有一定难度的书面作业，以锻炼学生运用本专业基本理论分析和解决实际问题的能力。期末考试采用开卷的形式，出题需要灵活有深度，以便能真正考核出学生的实际水平。

**五、教材、课程网址及参考书目**

教 材：《形式化方法导论》，张广泉。清华大学出版社，2015年12月。附加教师自编材料。

**执笔者：王婷**

**审核者：张繁**

**课程教学团队成员：王婷、陈铁明**